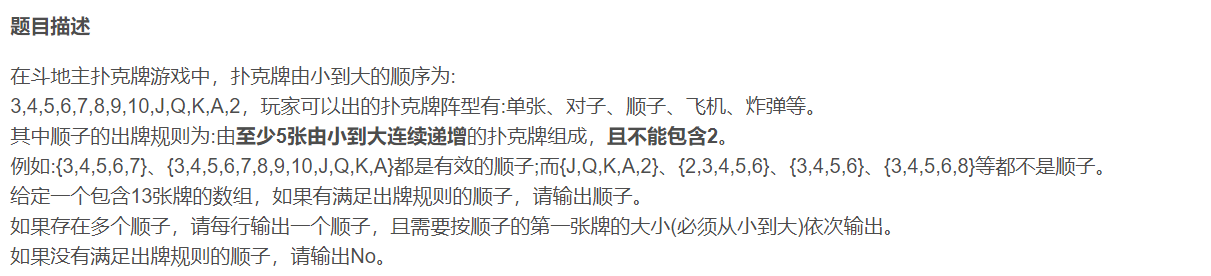
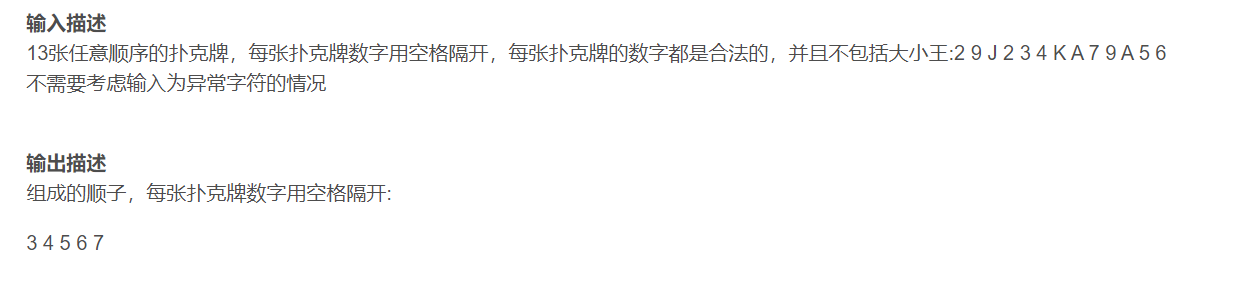
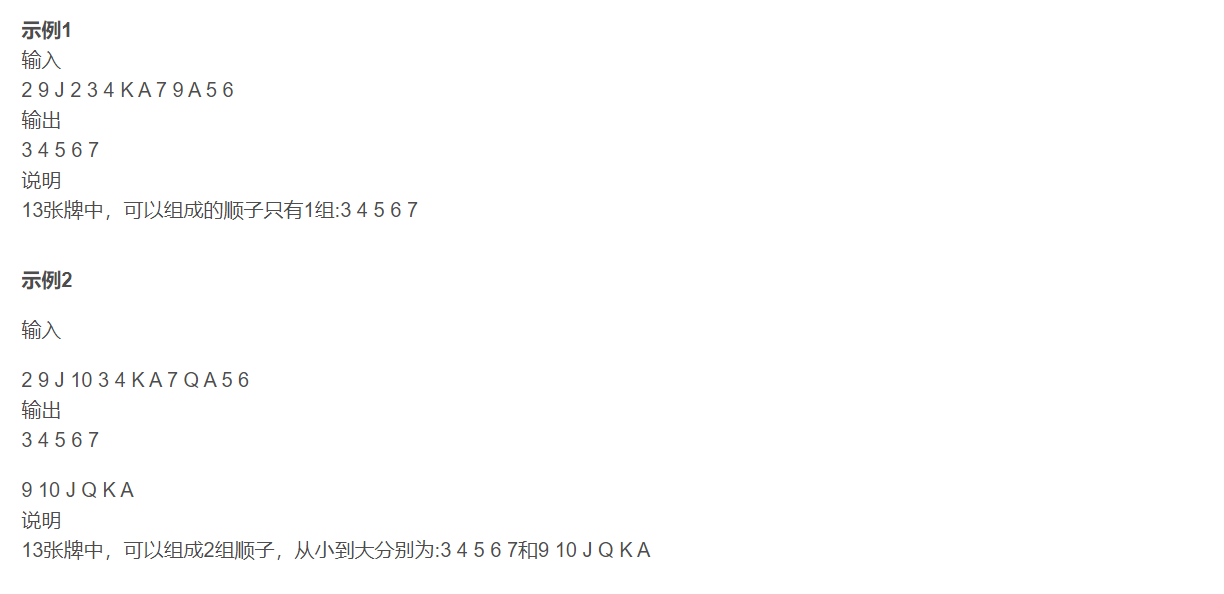
# **E卷-斗地主之顺子[100分]（ Java | Python3 | C++ | C语言 | JsNode | Go）**











import java.util.\*;

public class Main {

static String[] arr = {"3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"};

static int getID(String now) {

for (int i = 0; i < arr.length; i++) {

if (arr[i].equals(now)) {

return i;

}

}

return -1;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// 读取输入并将其分割成字符串数组

String[] s = scanner.nextLine().split(" ");

// 定义卡牌数组

// 使用集合来快速查找卡牌

Set<String> st = new HashSet<>(Arrays.asList(arr));

// 初始化计数数组

int[] c = new int[arr.length];

// 获取卡牌在数组中的索引

// 统计每种卡牌的数量

for (String x : s) {

int id = getID(x);

if (id != -1) {

c[id]++;

}

}

boolean ok = false;

// 检查是否存在连续的五张卡牌

for (int i = 0; i < 8; i++) {

boolean yes = true;

List<String> mp = new ArrayList<>();

for (int j = 0; j < 5; j++) {

if (c[i + j] == 0) {

yes = false;

break;

}

}

if (yes) {

for (int j = 0; j < 20; j++) {

if (i + j >= arr.length || c[i + j] == 0) {

break;

}

c[i + j]--;

mp.add(arr[i + j]);

}

// 打印找到的连续卡牌

System.out.println(String.join(" ", mp));

ok = true;

}

}

// 如果没有找到连续的五张卡牌，输出 "No"

if (!ok) {

System.out.println("No");

}

}

}



# 读取输入的一行字符串，并将其按空格分割成一个列表

s = input().split()

# 定义一个包含所有可能卡牌的列表

arr = ['3', '4', '5', '6', '7', '8', '9', '10', 'J', 'Q', 'K', 'A']

# 将卡牌列表转换成集合，用于后续的快速查找

st = set(arr)

# 初始化一个列表，用于计数每种卡牌的个数

c = [0] \* len(arr)

# 定义一个函数，根据卡牌的值找到其在 arr 列表中的索引

def getID(now):

for i in range(len(arr)):

if arr[i] == now:

return i

return -1

# 遍历输入的卡牌

for x in s:

# 如果卡牌在 arr 列表中，找到其索引并在计数列表中对应的位置加1

if getID(x) != -1:

c[getID(x)] += 1

# 初始化是否找到连续五张卡牌的标志

ok = 0

# 检查是否存在连续五张的卡牌组合

for i in range(0, 8):

yes = 1

mp = []

# 检查从当前位置开始的五张卡牌是否都存在

for j in range(0, 5):

if c[i + j] == 0:

yes = 0

break

# 如果存在连续五张的卡牌组合

if yes:

for j in range(0, 20):

# 如果索引超出列表长度或当前卡牌没有剩余就停止

if i + j >= len(arr) or c[i + j] == 0:

break

# 减少当前卡牌数量并将其加入结果集

c[i + j] -= 1

mp.append(arr[i + j])

# 打印找到的卡牌并设定标志

print(" ".join(mp))

ok = 1

# 如果没有找到任何连续五张的卡牌组合，输出 "No"

if not ok:

print("No")



#include <iostream>

#include <vector>

#include <string>

#include <set>

#include <algorithm>

using namespace std;

// 获取卡牌在数组中的索引

int getID(const string& now, const vector<string>& arr) {

for (int i = 0; i < arr.size(); i++) {

if (arr[i] == now) {

return i;

}

}

return -1;

}

int main() {

// 读取输入并将其分割成字符串数组

string input;

getline(cin, input);

vector<string> s;

size\_t pos = 0;

while ((pos = input.find(' ')) != string::npos) {

s.push\_back(input.substr(0, pos));

input.erase(0, pos + 1);

}

s.push\_back(input);

// 定义卡牌数组

vector<string> arr = {"3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"};

// 使用集合来快速查找卡牌

set<string> st(arr.begin(), arr.end());

// 初始化计数数组

vector<int> c(arr.size(), 0);

// 统计每种卡牌的数量

for (const string& x : s) {

int id = getID(x, arr);

if (id != -1) {

c[id]++;

}

}

bool ok = false;

// 检查是否存在连续的五张卡牌

for (int i = 0; i < 8; i++) {

bool yes = true;

vector<string> mp;

for (int j = 0; j < 5; j++) {

if (c[i + j] == 0) {

yes = false;

break;

}

}

if (yes) {

for (int j = 0; j < 20; j++) {

if (i + j >= arr.size() || c[i + j] == 0) {

break;

}

c[i + j]--;

mp.push\_back(arr[i + j]);

}

// 打印找到的连续卡牌

for (const string& card : mp) {

cout << card << " ";

}

cout << endl;

ok = true;

}

}

// 如果没有找到连续的五张卡牌，输出 "No"

if (!ok) {

cout << "No" << endl;

}

return 0;

}



#include <stdio.h>

#include <string.h>

// 获取卡牌在数组中的索引

int getID(char\* now, char arr[][3], int arr\_len) {

for (int i = 0; i < arr\_len; i++) {

if (strcmp(arr[i], now) == 0) {

return i;

}

}

return -1;

}

int main() {

char input[1000];

char\* s[100];

int s\_len = 0;

// 读取输入并将其分割成字符串数组

fgets(input, sizeof(input), stdin);

char\* token = strtok(input, " \n");

while (token != NULL) {

s[s\_len++] = token;

token = strtok(NULL, " \n");

}

// 定义卡牌数组

char arr[13][3] = {"3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"};

int arr\_len = 13;

// 初始化计数数组

int c[13] = {0};

// 统计每种卡牌的数量

for (int i = 0; i < s\_len; i++) {

int id = getID(s[i], arr, arr\_len);

if (id != -1) {

c[id]++;

}

}

int ok = 0;

// 检查是否存在连续的五张卡牌

for (int i = 0; i < 8; i++) {

int yes = 1;

char mp[20][3];

int mp\_len = 0;

for (int j = 0; j < 5; j++) {

if (c[i + j] == 0) {

yes = 0;

break;

}

}

if (yes) {

for (int j = 0; j < 20; j++) {

if (i + j >= arr\_len || c[i + j] == 0) {

break;

}

c[i + j]--;

strcpy(mp[mp\_len++], arr[i + j]);

}

// 打印找到的连续卡牌

for (int k = 0; k < mp\_len; k++) {

printf("%s ", mp[k]);

}

printf("\n");

ok = 1;

}

}

// 如果没有找到连续的五张卡牌，输出 "No"

if (!ok) {

printf("No\n");

}

return 0;

}



// 导入 readline 模块以便从标准输入读取数据

const readline = require("readline");

// 创建接口以读取标准输入

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

// 定义卡牌数组

const arr = ["3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"];

const arrLen = arr.length;

// 初始化计数数组

let c = new Array(arrLen).fill(0);

// 获取卡牌在数组中的索引

function getID(now) {

for (let i = 0; i < arrLen; i++) {

if (arr[i] === now) {

return i;

}

}

return -1;

}

// 读取输入并处理

rl.on("line", (input) => {

// 将输入分割成字符串数组

const s = input.trim().split(/\s+/);

// 统计每种卡牌的数量

s.forEach((x) => {

const id = getID(x);

if (id !== -1) {

c[id]++;

}

});

let ok = false;

// 检查是否存在连续的五张卡牌

for (let i = 0; i < 8; i++) {

let yes = true;

let mp = [];

for (let j = 0; j < 5; j++) {

if (c[i + j] === 0) {

yes = false;

break;

}

}

if (yes) {

for (let j = 0; j < 20; j++) {

if (i + j >= arrLen || c[i + j] === 0) {

break;

}

c[i + j]--;

mp.push(arr[i + j]);

}

// 打印找到的连续卡牌

console.log(mp.join(" "));

ok = true;

}

}

// 如果没有找到连续的五张卡牌，输出 "No"

if (!ok) {

console.log("No");

}

// 关闭 readline 接口

rl.close();

});



package main

import (

"bufio"

"fmt"

"os"

"strings"

)

// 获取卡牌在数组中的索引

func getID(now string, arr []string) int {

for i, v := range arr {

if v == now {

return i

}

}

return -1

}

func main() {

// 定义卡牌数组

arr := []string{"3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"}

arrLen := len(arr)

// 初始化计数数组

c := make([]int, arrLen)

// 读取输入并将其分割成字符串数组

scanner := bufio.NewScanner(os.Stdin)

scanner.Scan()

input := scanner.Text()

s := strings.Fields(input)

// 统计每种卡牌的数量

for \_, x := range s {

id := getID(x, arr)

if id != -1 {

c[id]++

}

}

ok := false

// 检查是否存在连续的五张卡牌

for i := 0; i < 8; i++ {

yes := true

var mp []string

for j := 0; j < 5; j++ {

if c[i+j] == 0 {

yes = false

break

}

}

if yes {

for j := 0; j < 20; j++ {

if i+j >= arrLen || c[i+j] == 0 {

break

}

c[i+j]--

mp = append(mp, arr[i+j])

}

// 打印找到的连续卡牌

fmt.Println(strings.Join(mp, " "))

ok = true

}

}

// 如果没有找到连续的五张卡牌，输出 "No"

if !ok {

fmt.Println("No")

}

}